AMENDMENTS TO THE CLAIMS:

This listing of claims replaces all prior versions and listings of claims in the application:

LISTING OF CLAIMS:

1. (Currently Amended) A method of modeling a logic design, comprising:

creating a graphical representation of the logic design;

receiving a selection that corresponds to a type of simulation code, the selection corresponding to one of plural different simulation languages;

generating simulation code based on the graphical representation and the selection, the simulation code comprising executable code in one of the plural different simulation languages that corresponds to the selection; and

using the simulation code to test operation of the logic design, wherein using the simulation code comprises:

propagating a state through the simulation code; and

determining if there is an error in the logic design based on a propagated state, wherein determining is performed via executable instructions that operate absent user intervention.

2. (Original) The method of claim 1, wherein the graphical representation is comprised of functional block diagrams and virtual wires that interconnect the functional block diagrams.

3. (Original) The method of claim 2, wherein creating comprises:

retrieving the functional block diagrams from a database; and

arranging the functional block diagrams and the virtual wires to model the logic design.

4. (Original) The method of claim 2, wherein creating comprises:

defining the functional block diagrams using simulation code; and

arranging the functional block diagrams and the virtual wires to model the logic design.

5. (Original) The method of claim 1, further comprising:

displaying a menu comprised of different types of functional block diagrams;

receiving an input selecting one of the different types of functional block diagrams;

retrieving a selected functional block diagram; and

creating the graphical representation of the logic design using the selected functional block diagram.

6. (Canceled)

7. (Previously Presented) The method of claim 1, wherein the state comprises one of a zero state, a one state, and an undefined state.

8. (Previously Presented) The method of claim 1, further comprising:

providing a visual indication if there is an error in the graphical representation of the logic design.

9. (Currently Amended) A method of modeling a logic design, comprising:

displaying a menu comprised of different types of functional block diagrams;

receiving an input selecting one of the different types of functional block diagrams;

receiving a selection that corresponds to a type of simulation code, the selection corresponding to one of plural different simulation languages;

retrieving a selected functional block diagram;

creating a graphical representation of a logic design using the selected functional block diagram, wherein creating comprises:

interconnecting the selected functional block diagram with one or more other functional block diagrams to generate a model of the logic design; and

defining the selected functional block diagram using the type of simulation code if a function of the selected functional block diagram is undefined when retrieved;

generating simulation code to simulate operation of the logic design based on the graphical representation and the selection, the simulation code comprising executable code in one of the plural different simulation languages that corresponds to the selection; and

using the simulation code to test the operation of the logic design, wherein using the simulation code comprises:

determining if there is an error in the logic design based on a propagated state, wherein determining is performed via executable instructions that operate absent user intervention.

10. (Canceled)

11. (Currently Amended) An article comprising a machine-readable medium that stores executable instructions for modeling a logic design, the instructions causing a machine to:

create a graphical representation of the logic design;

receive a selection that corresponds to a type of simulation code, the selection corresponding to one of plural different simulation languages;

generate simulation code based on the graphical representation and the selection, the simulation code comprising executable code in one of the plural different simulation languages that corresponds to the selection; and

use the simulation code to test operation of the logic design, wherein using the simulation code comprises:

propagating a state through the simulation code; and

determining if there is an error in the logic design based on a propagated state, wherein determining is performed via executable instructions that operate absent user intervention.

12. (Original) The article of claim 11, wherein the graphical representation is comprised of functional block diagrams and virtual wires that interconnect the functional block diagrams.

13. (Original) The article of claim 12, wherein creating comprises:

retrieving the functional block diagrams from a database; and

arranging the functional block diagrams and the virtual wires to model the logic design.

14. (Original) The article of claim 12, wherein creating comprises:

defining the functional block diagrams using simulation code; and

arranging the functional block diagrams and the virtual wires to model the logic design.

15. (Original) The article of claim 11, further comprising instructions that cause the machine to:

display a menu comprised of different types of functional block diagrams;

receive an input selecting one of the different types of functional block diagrams;

retrieve a selected functional block diagram; and

create the graphical representation of the logic design using the selected functional block diagram.

16. (Canceled)

17. (Previously Presented) The article of claim 11, wherein the state comprises one of a zero state, a one state, and an undefined state.

18. (Previously Presented) The article of claim 11, further comprising instructions that cause the machine to:

provide a visual indication if there is an error in the graphical representation of the logic design.

19. (Currently Amended) An article comprising a machine-readable medium that stores executable instructions that cause a machine to:

display a menu comprised of different types of functional block diagrams;

receive an input selecting one of the different types of functional block diagrams;

receive a selection that corresponds to a type of simulation code, the selection corresponding to one of plural different simulation languages;

retrieve a selected functional block diagram;

create a graphical representation of a logic design using the selected functional block diagram, wherein creating comprises:

interconnecting the selected functional block diagram with one or more other functional block diagrams to generate a model of the logic design; and

defining the selected functional block diagram using the type of simulation code if a function of the selected functional block diagram is undefined when retrieved;

generate simulation code to simulate operation of the logic design based on the graphical representation and the selection, the simulation code comprising executable code in one of the plural different simulation languages that corresponds to the selection; and

use the simulation code to test the operation of the logic design, wherein using the simulation code comprises:

determining if there is an error in the logic design based on a propagated state, wherein determining is performed via executable instructions that operate absent user intervention.

20. (Canceled)

21. (Currently Amended) An apparatus for modeling a logic design, comprising:

memory that stores executable instructions; and

a processor that executes the instructions to:

create a graphical representation of the logic design;

receive a selection that corresponds to a type of simulation code, the selection corresponding to one of plural different simulation languages;

generate simulation code based on the graphical representation and the selection, the simulation code comprising executable code in one of the plural different simulation languages that corresponds to the selection; and

use the simulation code to test operation of the logic design, wherein using the simulation code comprises:

propagating a state through the simulation code; and

determining if there is an error in the logic design based on a propagated state, wherein determining is performed via executable instructions that operate absent user intervention.

22. (Original) The apparatus of claim 21, wherein the graphical representation is comprised of functional block diagrams and virtual wires that interconnect the functional block diagrams.

23. (Original) The apparatus of claim 22, wherein creating comprises:

retrieving the functional block diagrams from a database; and

arranging the functional block diagrams and the virtual wires to model the logic design.

24. (Original) The apparatus of claim 22, wherein creating comprises:

defining the functional block diagrams using simulation code; and

arranging the functional block diagrams and the virtual wires to model the logic design.

25. (Original) The apparatus of claim 21, wherein the processor executes instructions to:

display a menu comprised of different types of functional block diagrams;

receive an input selecting one of the different types of functional block diagrams;

retrieve a selected functional block diagram; and

create the graphical representation of the logic design using the selected functional block diagram.

26. (Canceled)

27. (Previously Presented) The apparatus of claim 21, wherein the state comprises one of a zero state, a one state, and an undefined state.

28. (Previously Presented) The apparatus of claim 21, wherein the processor executes instructions to:

provide a visual indication if there is an error in the graphical representation of the logic design.

29. (Currently Amended) An apparatus comprising:

memory that stores executable instructions; and

a processor that executes the instructions to:

display a menu comprised of different types of functional block diagrams;

receive an input selecting one of the different types of functional block diagrams;

receive a selection that corresponds to a type of simulation code, the selection corresponding to one of plural different simulation languages;

retrieve a selected functional block diagram; and

create a graphical representation of a logic design using the selected functional block diagram, wherein creating comprises:

interconnecting the selected functional block diagram with one or more other functional block diagrams to generate a model of the logic design; and

defining the selected functional block diagram using the type of simulation code if a function of the selected functional block diagram is undefined when retrieved;

generate simulation code to simulate operation of the logic design based on the graphical representation and the selection, the simulation code comprising executable code in one of the plural different simulation languages that corresponds to the selection; and

~~using~~ use the simulation code to test the operation of the logic design, wherein using comprises:

determining if there is an error in the logic design based on a propagated state, wherein determining is performed via executable instructions that operate absent user intervention.

30. (Canceled)